
Sun
microsystems

Web Services: J2EE™ vs .NET

"You make the call..."

Reginald R. Hutcherson
Manager, Technology Evangelism & Adoption
Sun Microsystems Inc.

Learning Objectives

- Understand web services
- Understand the J2EE™ platform and .NET
- Cut through the FUD
  - Fear, Uncertainty, and Doubt
- Empower you to make the right decision!

Presentation Roadmap

- **Web services fundamentals**
- Under The Hood: J2EE vs .NET
- Comparisons: J2EE vs .NET
- Conclusions

Building Web Services

- **Really XML Interfaces**
  - Application, Systems, Services are Old!
- **Provider**: Creates, Assembles & Deploys
  - Old Technology: J2EE, Middleware, etc.
- **Provider**: Describes with WSDL...others
- **Provider**: Registers with UDDI...others
- **USER**: Searches UDDI & Binds
  - SOAP, ebXML

Web Services Approach

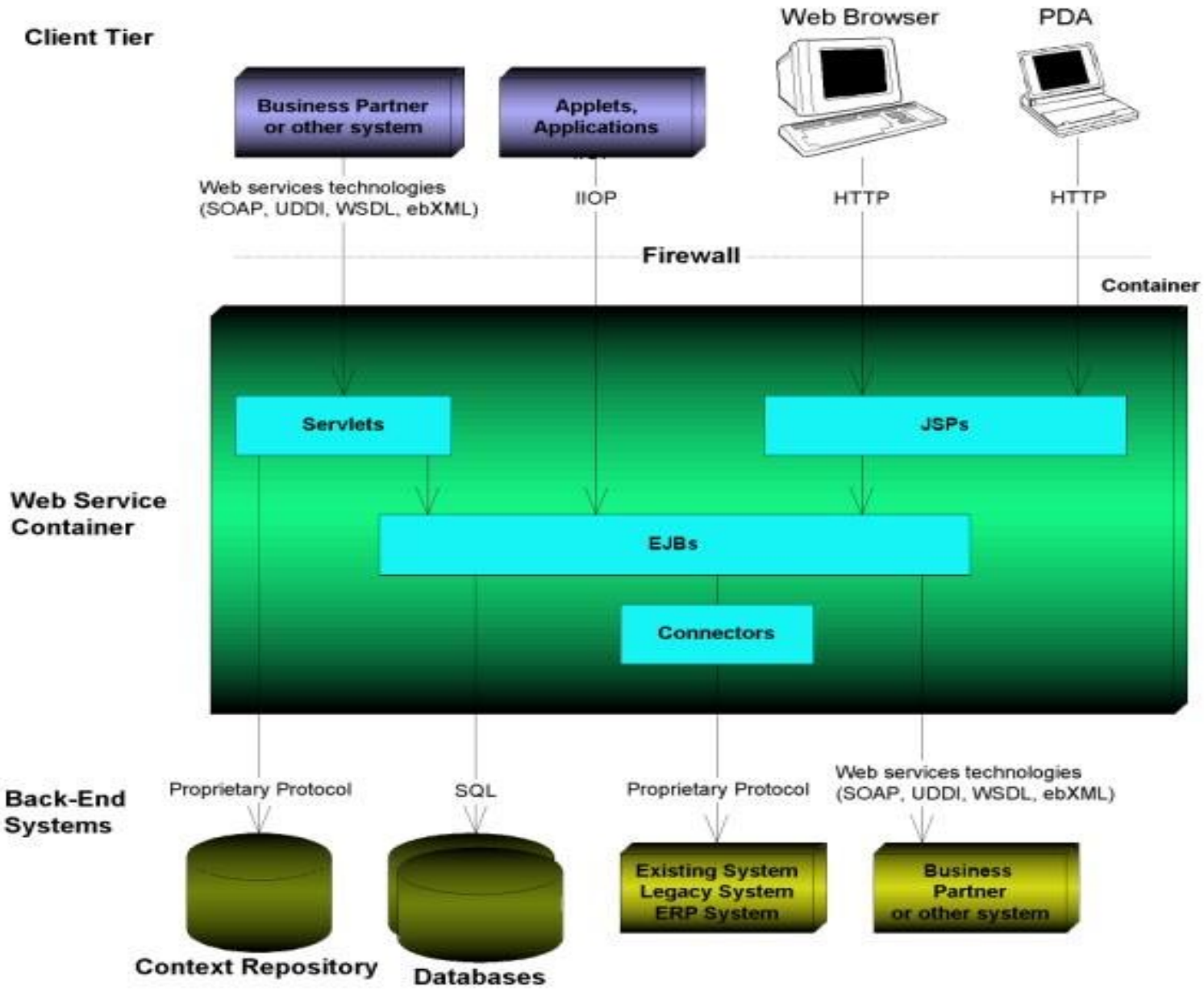
- **J2EE & .NET**
  - Repositioned for Web Services
- **Plumbing**:
  - XML Interoperability, Load Balancing
  - Transaction management
- **Use Containers** – Don’t do it yourself!

Presentation Roadmap

- Web services fundamentals
- **Under The Hood: J2EE vs .NET**
- Comparisons: J2EE vs .NET
- Conclusions

Under The Hood: J2EE...

- **Java The Foundation**
  - Language, JVM & Platform
- **J2EE & WebServices**
  - Enterprise Services, includes XML
- **Container Services**
- **Business Layer**
  - Existing: EJB, JDBC, JCA
  - New Stuff: SOAP, UDDI, WSDL & ebXML


Client Tier
Web Browser
PDA
Business Partner or other system
Applets, Applications
Web services technologies (SOAP, UDDI, WSDL, ebXML)
IIOP
HTTP
HTTP
Firewall
Container
Servlets
JSPs
Web Service Container
EJBs
Connectors
Back-End Systems
Proprietary Protocol
SQL
Proprietary Protocol
Web services technologies (SOAP, UDDI, WSDL, ebXML)
Existing System Legacy System ERP System
Business Partner or other system
Context Repository
Databases

J2EE Platform Summary

- Industry standard
- 50+ vendors implement the standard (tools, app servers, etc.)
- Result of collaboration between vendors
- The J2EE platform Includes:
  - PDF file agreements
  - Reference Implementation
  - Test Suite
  - J2EE Blueprints design guidelines
- Based on Java™ technology—JRE interprets bytecode

Under The Hood: .NET...

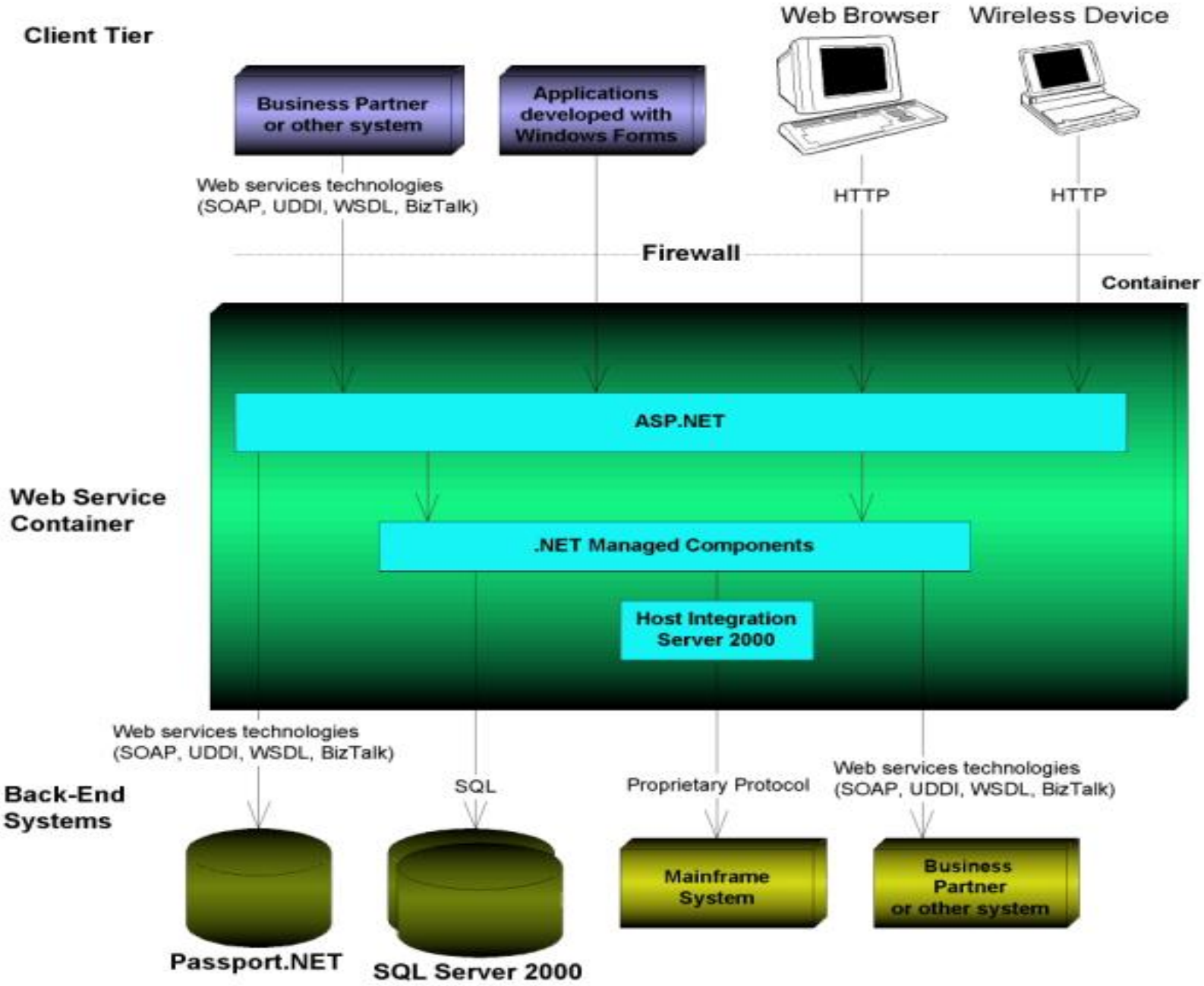
- Microsoft .NET
  - Product suite, building web services
- Rewrite of Windows DNA
  - MTS, COM+, MSMQ, SQL Server
- .NET Includes
  - New Web Services & Language Support

Under The Hood: .Net...

- Language Independence & Interoperability
  - VB.NET & C# (OO Language)
- Intermediate Language
  - Analogous to Java Bytecode!
- Common Language Runtime (CLR)
  - Analogous to Java JVM!
- Provides: GC, Exception Handing, etc.

Under The Hood: .NET...

- **Hailstorm**
  - Shared Context over Web Services
  - Passport for Identity, Security, etc.
  - For Fee & Free Services
- **Resolves:** "Islands" of information


Client Tier
Web Browser
Wireless Device
Business Partner or other system
Applications developed with Windows Forms
Web services technologies (SOAP, UDDI, WSDL, BizTalk)
HTTP
HTTP
Firewall
Container
ASP.NET
Web Service Container
.NET Managed Components
Host Integration Server 2000
Web services technologies (SOAP, UDDI, WSDL, BizTalk)
SQL
Proprietary Protocol
Web services technologies (SOAP, UDDI, WSDL, BizTalk)
Back-End Systems
Passport.NET
SQL Server 2000
Mainframe System
Business Partner or other system

.NET Summary

- MSFT product suite (not standard)
- Previous version was Windows DNA
  - Included COM+, MSMQ, etc.
- .NET Framework replaces Windows DNA
- .NET Servers (SQL Server, BizTalk Server, etc.)
- Hailstorm Services, Passport.NET
- Visual Studio.NET
- Common Language Runtime (CLR)
  - Provides language neutrality through IL code
- New C# language

J2EE and .NET Analogies

Feature	J2EE	.NET
Type of Technology	Standard	Product
Middleware Vendors	30+	Microsoft
Interpreter	JRE	CLR
Dynamic Web Pages	JSP™	ASP.NET
Middle−Tier Components	EJB	.NET Managed Components
Database Access	JDBC, SQL/J	ADO.NET
SOAP, WSDL, UDDI	Yes	Yes
Implicit Middleware (load−balancing, etc.)	Yes	Yes

Presentation Roadmap

- Web services fundamentals
- Under The Hood: J2EE vs .NET
- **Comparisons: J2EE vs .NET**
- Conclusions

RAD Features

- **The J2EE platform**
  - State management
  - Persistence services (entity beans)
  - Custom JSP™ tag libraries
  - Proprietary features
- **.NET**
  - Queued Components
  - Client device independence tools
  - Business Process Management and E−Commerce
- **Conclusion:** Comparable

Performance Features

- **Problem**
  - Backend Integration: Databases
  - Enterprises: Data driven not logic driven
- **Solutions**
  - Reduce data integration
  - Provide access to tactics

Performance Features

- **J2EE**
  - Low−level database integration
  - Stateful processes
  - Long−term caching
- **.NET**
  - Backend Integration hidden
- **Conclusion**: J2EE

Single Vendor Solution

- **J2EE Platform**
  - **Good**: Large players single−vendor solution
  - **Bad**: Small vendors mix−and−match
- .**NET**
  - **Good**: Get most of what you need from MSFT
  - **Bad**: API is not open, hard to create new tools
- **Legacy Integration**
  - **J2EE** platform offers single−vendor solution
    - Legacy IBM, BEA, Oracle, etc customers
  - .**NET** offers single−vendor solution
    - Legacy MSFT customers
- **Conclusion**: J2EE

Support for Existing Systems

- **Problem**
  - Heterogeneous environments
  - Strategy for preserving & reusing investments
- **Solution**
  - Open integration solutions
  - Packaged applications & legacy systems

Support for Existing Systems

- **J2EE Platform**
  - JMS (Java™ Message Service API)
  - Web services
  - CORBA
  - JNI (Java Native Interface)
  - J2EE Connector Architecture (**JCA**)
- **.NET**
  - Host Integration Server 2000
  - COM TI (Mainframes)
  - MSMQ (MQSeries)
  - BizTalk (B2B Protocols, EDI)

Support for Existing Systems

- J2EE
  - **Good**: Integration vision through JCA
  - **Bad**: JCA adapters are not a reality today
  - **Promising**: Tremendous buy−in
- .NET
  - **Good**: .NET decent legacy story today
  - **Bad**: Vision for tomorrow is horrible
- **Conclusion**: J2EE

Market Perception

- **Problem**
  - Huge switching cost!
  - Integrates with new technology (web services)
- **Solution**
  - Leverage
    - Existing investments
    - Reduce switching cost & training
    - Extend legacy & web services

Market Perception

- **J2EE Platform**
  - **Good**: Marketing by 50+ vendors
  - **Bad**: Perceived as enterprise platform
- **.NET**
  - **Good**: Marketing by 1 vendor
  - **Good**: Good job on perception so far
  - **Good**: Perceived web services platform
- **Conclusion**: .NET

Architecture Maturity

- **J2EE Platform**
  - Platform mature—Web services are new
- **.NET**
  - Almost a complete rewrite of DNA
  - CLR, C# is brand new
- **Conclusion**
  - Good: J2EE evolution of stable software
  - Risky: .NET in beta now; First generation
  - Risky: Migration is also challenging for .NET

Language Support

- **J2EE Platform**
  - Bet on Java language
  - Bridge to other languages (CORBA, JNI, Connectors, web services)
- **.NET**
  - Language neutral through CLR
  - Can inherit from different languages
  - No need to bridge as in the J2EE platform

Language Support

- .NET mixed language in CLR warnings
  - Risky to disrupt existing systems
  - Hard to maintain mixed code
  - Hard to share best practices and communicate
  - Knowledge split means if someone leaves, code may be not understandable

Language Support

- CLR is good for single−language solutions
  - Compelling for existing legacy code to become .NET code
- There are issues though
  - Retraining OO
  - Bound code to .NET
  - Why invest in outdated technologies?
- **Conclusion**
  - We would rather bridge to existing systems
  - Not a huge value that CLR is language neutral
  - J2EE once again..

Portability Features

- The J2EE platform
  - Hardware−agnostic
  - Middleware−agnostic, enforced by test suite
  - Problem: Test suite does not cover web services
  - Portability is, though, better than MSFT
- .NET
  - Win32 only
- Conclusions
  - The J2EE platform wins on portability

Web Services Support

- The J2EE platform
  - JAXP (Java™ API for XML parsing) available now
  - Can deploy today
  - RAD development tools through 3rd parties
  - Downside: Other JAX APIs not done yet
- .NET
  - Pros: RAD development of web services through Visual Studio.NET—Awesome tool
  - Downsides: No ebXML, can't deploy today

Shared Context Support

- **J2EE Platform**
  - Distributed repository vision
  - JDBC™ API, future shared context APIs
  - Pros: No big brother effect, no single point of failure, more scalable to needs of masses
- **.NET**
  - Single repository (Passport.NET)
  - Issue: Huge Lock−In potential..
- Conclusions
  - Still too early to tell

System Cost

- **J2EE Platform**
  - Can choose your service level
  - High−end (BEA, IBM, Oracle)
  - Almost free (jBoss/Cobalt/Linux)
  - Get what you pay for
- **.NET**
  - Cheap deployment (comes with OS)
  - Maintenance issues with lower−end systems
- **Conclusions**
  - Both systems can be low cost
  - Consider TCO, not cost per transaction

Presentation Roadmap

- Web services fundamentals
- Under The Hood: J2EE vs .NET
- Comparisons: J2EE vs .NET
- **Conclusions**

J2EE and .NET Summarized

Platform Feature	J2EE	.NET
RAD	✔	✔
Performance	✔	✔
Single Vendor	✔	✔
Interoperability	✔	✔
Marketing	✔	✔
Maturity	✔	✔
Language Support	✔	✔
Portability	✔	✔
Web Service Support	✔	✔
Total Cost Ownership	✔	✔
Summary	✔	


Sun
microsystems